第27卷 第2期 工 程 数 学 学 报 Vol. 27 No. 2
2010年04月 CHINESE JOURNAL OF ENGINEERING MATHEMATICS Apr. 2010

文章编号:1005-3085(2010)02-0219-06

多块结构化网格CFD并行计算和负载平衡研究*

郑秋亚[1,2]， 刘三阳[1]， 左大海[2]， 梁益华[3]

(1- 西安电子科技大学理学院，西安 710071； 2- 长安大学理学院，西安 710064；
3- 中国航空计算技术研究所航空流体动力数值模拟重点实验室，西安 710068)

摘 要: 基于连续拼接多块结构化网格，通过求解雷诺平均Navier-Stokes方程研究并行计算中的负载平衡问题。利用组合优化中的排序理论设计负载平衡算法，实现了网格数据的自动划分和各处理机上计算任务的自动分配。在工作站集群MPI并行环境下，通过实例考察了负载平衡算法和并行计算的性能，16个处理机上的负载均方差和负载相对均方差分别为0.0084和0.1347%，并行计算结果和实验数据吻合良好，并行效率高。本文算法具有良好的可扩展性，适用于MIMD结构计算机上基于多块结构化网格并行计算中的负载平衡问题。

关键词: Navier-Stokes方程；负载平衡；并行计算；排序

分类号: AMS(2000) 76M12　　**中图分类号**: V211.3　　**文献标识码**: A

1 引言

新型飞行器的气动设计和研制要求对具有分离、旋涡、激波/附面层干扰等复杂流动特性的气动问题采用雷诺平均Navier-Stokes (RANS) 方程，求解这类方程往往要求计算达到千万网格点的规模。对于这种大规模高精度的流体力学计算问题，人们常利用高性能计算机进行并行计算以提高计算速度。并行计算作为CFD研究的一个主要方向，得到了国内外研究人员的广泛重视和快速发展[1-3]。

区域分解算法将问题的求解区域划分成多个子区域，这些子区域相互包含相邻区域的拟边界信息，相互迭代共同求解同一问题。并行计算与区域分解算法相结合，将网格数据分配给多个处理机完成，计算过程中利用PVM (Parallel Virtual Machine) 或MPI (Message Passing Interface) 消息传递接口处理机之间进行适当通信。对于这种基于消息传递模式的大粒度数据并行来说，影响并行效率的主要因素主要是负载平衡。

本文采用分区策略，根据计算模型几何上的特性进行分区，生成初始计算网格。将并行计算中的负载平衡问题化成极小化最大完工时间的可拆分排序问题，设计负载平衡算法，实现网格数据的自动划分和计算任务到各处理机上的自动分配，并通过实例对本文负载平衡算法的性能和并行算法的可扩展性进行评估。

2 控制方程和数值方法

控制方程为无量纲化时间相关的三维守恒型RANS方程，在一般曲线坐标系下其形式为

收稿日期: 2009-05-25. **作者简介**: 郑秋亚 (1964年11月生)，女，博士，高级工程师. 研究方向：计算流体力学.
***基金项目**: 航空科学基金 (20081431)；国家自然科学基金 (60974082).

$$\frac{\partial Q}{\partial t}+\frac{\partial E}{\partial \xi}+\frac{\partial F}{\partial \eta}+\frac{\partial G}{\partial \zeta}=\frac{1}{R_e}\Big[\frac{\partial E_v}{\partial \xi}+\frac{\partial F_v}{\partial \eta}+\frac{\partial G_v}{\partial \zeta}\Big], \tag{1}$$

其中 Q 是守恒变量，E,F 和 G 为无粘通矢量，E_v，F_v 和 G_v 为粘性通矢量。

数值方法采用 Jameson[4] 中心格式加自适应二阶和四阶人工粘性，用隐式 LU-SGS 方法[5] 沿时间推进，湍流模型为 Spalart-Allmaras (S-A) 一方程湍流模型[6]。

由于使用隐式时间向前推进方法，各子区域边界点上的计算需要用到其相邻子区域边界点上的数据，为了使各子区域上的隐式求解互不相关，对网格块“内边界”进行显式处理。如图1所示，A,B 为相邻子域，其边界面上的网格连续拼接，若 A 区域上的数值解用符号 $u_j^n\,(j=0,1,2,\cdots,m)$ 标记，B 区域上的数值解用符号 $v_j^n\,(j=0,1,2,\cdots)$ 标记，那么边界显式处理的结果为：$v_0^{n+1}=u_{m-1}^n$，$u_m^{n+1}=v_1^n$，这样处理并不会影响计算方法的稳定性，且能得到稳定收敛的结果[7-8]。

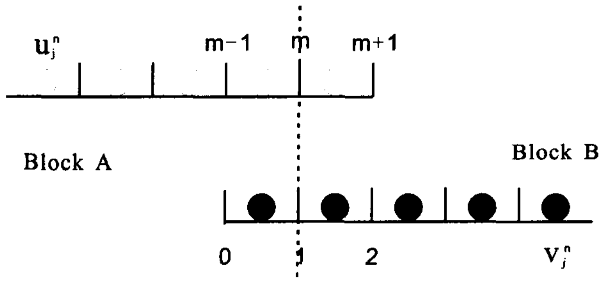

图 1: 拼接网格

3 负载平衡算法

负载平衡实际上是在并行机或分布式多处理机的各节点机之间重新分配工作量的一个过程。其目的是设计一种排序算法实现计算任务到各节点机上的分配，使得尽可能快地完成全部计算任务。对于基于消息传递的数据并行模式来说，负载平衡算法一般包括数据划分和任务到处理机上分配两个方面。

3.1 模型

设有 m 个处理机，处理机 M_i 的计算速度 (每秒浮点运算次数) 为 s_i，n 块网格，网格 G_j 所包含的网格单元数为 N_j，允许网格拆分成多个子网格在不同的处理机上进行计算，但同一网格块的任何两个不同的子网格不能在同一个处理机上计算。为了确保整个计算的稳定性和可实现性，要求网格拆分的次数尽量少，拆分后所得子网格块的规模不能太小。如何将这 n 块网格分配给 m 个处理机才能使整个计算任务在最短时间内完成。这属于极小化最大完工时间的可拆分平行机排序问题。假设：不计边界单元上的计算量和处理机之间的通信时间，若向前推进1个时间步，每个网格单元上需要执行的浮点运算次数为 W 次，则所有网格单元上需要执行的浮点运算次数之和为 $\sum\limits_{j=1}^{n} N_j\times W$，全部处理机的计算速度之和为 $\sum\limits_{i=1}^{m} s_i$，那么全部处理机共同完成所有计算任务需要的平均时间为

$$T_{\text{ave}}=\Big[\sum_{j=1}^{n}N_j\times W\Big]\Big/\sum_{i=1}^{m}s_i. \tag{2}$$

若用 G_i 表示分配到处理机 M_i 上的网格块集合，则处理机 M_i 完成其上任务需要的时间为

$$t_i = \Big[\sum_{k\in G_i} N_k \times W\Big]/\,s_i, \tag{3}$$

目标为

$$\min \max_{1\le i\le m} t_i. \tag{4}$$

3.2 算法

为避免网格拆分过程中出现过小的子网格块，本文在算法中设置一较小量 ε。

1) 计算所有处理机完成全部计算任务的平均时间 $T_{\rm ave}$；
2) 将未被分配的网格块按加工时间不增排序建立链表 1；
3) 当链表 1 非空时；
   3.1) 找出当前速度最快的处理机 M_i；
   3.2) 将链表 1 当前位置处的网格块 G_j 加载到 M_i 上求出 M_i 的完工时间 t_i；
   3.3) 如果 $t_i > T_{\rm ave}(1+\varepsilon)$，将网格块 G_j 按一定比例剖分成两个子网格 G_j^1 和 G_j^2，使得 G_j^1 分给处理机 M_i 后，M_i 的完工时间约为 $T_{\rm ave}$，修改子网格块 G_j^1 和 G_j^2 的边界条件；
   3.4) 将子网格 G_j^1 分配给处理机 M_i，将子网格 G_j^2 插入到链表 1 中，直到链表 1 为空。

4 实例及结果分析

以绕全机多块结构化网格为例，同时引入负载均方差和负载相对均方差指标来测评本文负载平衡算法的性能[9]。一般来说，负载均方差和负载相对均方差越小说明各处理机上的负载越均衡。以非对称 $C6$ 弹体为例评估并行计算结果的可信度和并行算法的可扩展性。

4.1 带双立尾全机

初始网格为 26 块结构网格，其规模见图2，图3是双立尾全机表面网格。机器是16个处理机构成的工作站集群，其中第 5、7、9 号节点机的浮点运算速度是其余节点机的 1.25 倍，16 个处理机的平均运算速度为 16.1875 mflops。

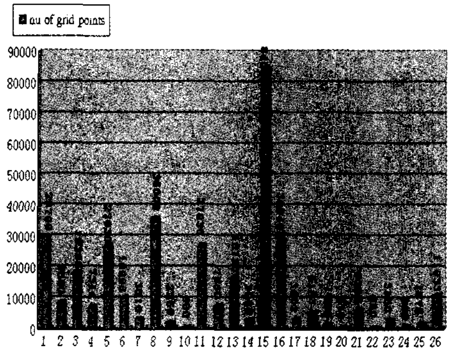

图 2: 26 块网格的规模分布

图 3: 带双立尾全机表面网格

假设沿时间每向前推进 1 个时间步，每个网格单元上需要执行 4810 次浮点运算数。可求得每向前推进 1 个时间步 16 个处理机的平均完工时间为 6.23799 秒，取 $\varepsilon = 0.05$，实施本文负载

平衡算法，26 块网格被拆分 9 次最终变成为 35 块网格，35 块网格在 16 个节点机上的分配情况和各节点机的预计完工时间见表 1。

表 1: 各节点机上的任务分配和完工时间

节点	任务序列	完工时间	节点	任务序列	完工时间	节点	任务序列	完工时间
1	28	6.25625	7	27	6.32500	13	26-12-23-10	6.15175
2	16	6.25623	8	3-9-35	6.27275	14	31-34-14-20	6.07612
3	1	6.31125	9	13-7-24	6.11531	15	6-33-18	6.42950
4	5	6.22050	10	8	6.24442	16	32-2-17-22	6.06925
5	15	6.27080	11	30-21	6.29750			
6	11	6.22050	12	29-4-25-19	6.26004			

表 2 是本文负载平均算法的性能指标，从表 2 可以看出，每向前推进 1 个时间步需要花费的计算机机时为 6.4295 秒，是机器平均完工时间的 1.0307 倍，16 个节点机的负载均方差和负载相对均方差分别为 0.0084 和 0.1347%。结果表明，本文负载平衡算法性能良好，用此算法分配到各节点机上的负载相当均衡。

表 2: 负载平衡算法的性能指标

处理机数	平均时间	最大完工时间	负载均方差	载相对均方差 (%)
16	6.23799	6.42950	0.0084	0.1347

负载均方差和负载相对均方差分别定义为

$$l_\sigma = \sum_{i=1}^{m}(l_i - l_{\text{ave}})^2/m, \quad l_{r\sigma} = l_\sigma/l_{\text{ave}}, \tag{5}$$

其中 l_i 表示第 i 个节点机上的负载量，l_{ave} 表示平均负载量，m 表示处理机个数。

4.2 非对称 $C6$ 弹体

采用大小为 $81 \times 43 \times 49$ 的两块结构网格，在由 8 台 CPU 速度为 2.0 GHz 的计算机构成的并行环境下 (其中主节点的内存为 2Gbytes，其他 7 个节点机的内存为 1Gbytes) 对绕 $C6$ 弹体的流场进行 RANS 方程并行计算，计算状态为 Mach=0.9，$\alpha = 0^0$ 和 10^0。图 4 是 $C6$ 弹体的 $C-O$ 型计算网格拓扑结构，图 5 和图 6 是使用 8 台机器进行并行计算所得弹体对称面压力系数分布与实验数据的比较。从图 5 和图 6 可以看出，并行计算结果和实验数据吻合良好，说明本文并行计算程序、通信和计算结果均可信。

4.3 算法的可扩展性

并行算法的可扩展性 (Scalability) 是指并行计算的性能随处理机数目的增加而按比例提高的能力。依据等效率度量法[10]，随着处理机数目的增加，若需要增加较少量的计算规模，就可使并行计算效率保持不变，那么该并行算法具有良好的扩展性。

为测评本文并行算法的可扩展性，在由 8 台 CPU 速度为 2.0 GHz 的计算机构成的并行环境下，以 $C6$ 弹体规模为 $2 \times 81 \times 43 \times 49$ 的计算网格和 2 个处理机为起点，使网格点数和处理机数目均成倍增加。测得 20 步的并行计算时间和并行效率见表 3。从表 3 可以看出，粗、中、细网格分别在 2、4、8 台机器上的并行效率分别为 94.81%, 92.74% 和 91.93%。随着机器数目的

增加，并行效率稍有减少。对于这种并行效率小幅降低现象，我们认为，主要是由于随着处理机个数的增加，网格拆分的次数相应增多，处理机之间的通信量相应增大，而并行计算机的通信能力未得到相应提高所致。依据等效率度量法，本文的并行算法具有较好的可扩展性。

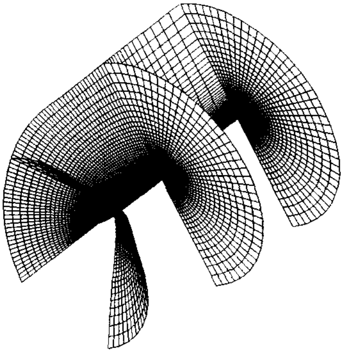

图 4: $C6$弹体网格结构

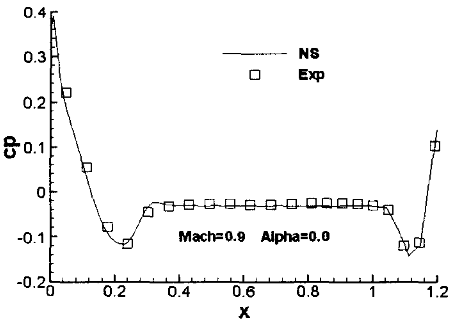

图 5: 对称面压力系数分布

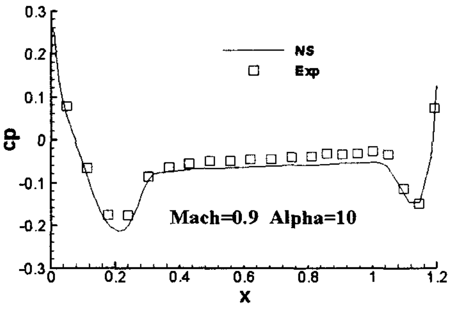

图 6: 对称面压力系数分布

表 3: 并行效率

网 格	单机	2 节点	并行效率 (%)	4 节点	并行效率 (%)	8 节点	并行效率 (%)
粗/172032	162.55	85.72	94.81	47.38		24.43	
中/344064	326.69	175.76		88.07	92.74	47.38	
细/688128	687.77	350.43		183.61		93.52	91.93

5 结论

利用组合优化中的排序理论设计了一种负载平衡算法，实现了网格数据的自动划分和各处理机上任务的自动分配。带双立尾全机的结果表明，本文负载平衡算法可行、性能良好。非对称$C-6$弹体结果表明，本文并行算法具有较好的可扩展性。本文负载平衡算法适用于 MIMD 结构计算机上基于多块结构化网格并行计算中的负载平衡问题。

参考文献:

[1] Roose D, Driessche R. Parallel computers and parallel algorithms for CFD: an introduction[R]. AGARD Report, R-807, 1995

[2] 朱国林，徐庆新. 计算流体力学并行计算技术研究综述[J]. 空气动力学学报，2002, 20: 1-6
Zhu G L, Xu Q X. Review on parallel computation technique on computational fluid dynamics in CAI[J]. ACTA Aerodynamica Sinica, 2002, 20: 1-6

[3] 陈国良，孙广中，徐云等. 并行算法研究方法学[J]. 计算机学报，*2008, 31(9)*: 1493-1502
Chen G L, Sun G Z, Xu Y, *et al*. Methodology of research on parallel algorithms[J]. Chinese Journal of Computers, 2008, 31(9): 1493-1502

[4] Jameson A, Schmidt W, Tarhel E. Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes[R]. AIAA, 1981-1259

[5] Yoon S, Jameson A. A lower-upper Symmetric-Gauss-Seidel method for the Euler and Navier-Stokes equations[R]. AIAA, 1987-0600

[6] Spalart P R, Allmaras S R. A one-equation turbulence model for aerodynamic flows[R]. AIAA, 1992-0439

[7] 吴建平，李晓梅. 三维Euler方程组隐式LU分解的高性能并行计算[J]. 空气动力学学报，2007, 25(4): 437-442
Wu J P, Li X M. High performance parallel computing of implicit LU scheme for 3D Euler equations[J]. ACTA Aerodynamica Sinica, 2007, 25(4): 437-442

[8] 庄礼深，吴子牛. 流体力学分区算法及相关理论问题[J]. 计算物理，2006, 23: 253-265
Zhuang L S, Wu Z N. Multiblock method and related theories for computational fluid dynamics[J]. Chinese Journal of Computational Physics, 2006, 23: 253-265

[9] 向建军，白欣，左继章. 一种用于实时集群的多任务负载均衡算法[J]. 计算机工程，2003, 29(12): 36-38
Xiang J J, Bai X, Zuo J Z. A multipletask load balancing algorithm used in real-time cluster system[J]. Computer Engineering, 2003, 29(12): 36-38

[10] 陈国良. 并行计算—结构—算法—编程[M]. 北京：高等教育出版社，2002
Chen G L. Parallel Computing: Architecture, Algorithm, Programming[M]. Beijing: Higher Education Press, 2002

CFD Parallel Computing and Load Balancing Research Using Multi-block Structured Grids

ZHENG Qiu-ya[1,2], LIU San-yang[1], ZUO Da-hai[2], LIANG Yi-hua[3]

(1- School of Science, Xidian University, Xi'an 710071; 2- School of Science, Chang'an University, Xi'an 710064; 3- Aeronautical Key Laboratory of Computational Fluid Dynamics, Aeronautics *Computing Technique Research Institute of China, Xi'an 710068)*

Abstract: This paper discusses the parallel computing and load balancing problems of Navier-Stokes equations, which are solved by using the multi-block structured grids with grid points matched on sub-domain boundaries. A load balancing algorithm is developed by using the scheduling method in combinatorial optimization, so that the grid partitioning and the task allocation can be implemented automatically. Several cases are performed on an MPI-based workstation cluster to investigate the performance of the load balancing algorithm and the parallel computing. With 16 processors, the absolute and relative load mean squared deviation are 0.0084 and 0.1347%, respectively. The computed results agree well with the experiment, and the parallel efficiency is high. This algorithm has a good scalability and can be used to deal with load balancing problems of CFD parallel computing which uses the multi-block structured grids on the MIMD computer system.

Keywords: Navier-Stokes equations; load balancing; parallel computing; scheduling

Received: 25 May 2009. **Accepted:** 16 Dec 2009.

Foundation item: The Aviation Science Foundation of China (20081431); the National Natural Science Foundation of China (60974082).